1. **NullPointerException (NPE)**: Essa exceção ocorre quando você tenta acessar um método ou propriedade de um objeto que é null, ou seja, não está apontando para uma instância válida. Isso pode acontecer quando você tenta acessar um membro de um objeto que não foi inicializado corretamente.
2. **ArrayIndexOutOfBoundsException**: Essa exceção ocorre quando você tenta acessar um índice inválido em um array. Por exemplo, tentar acessar um índice negativo ou um índice maior que o tamanho do array.
3. **ArithmeticException**: Essa exceção é lançada quando ocorre um erro em uma operação aritmética, como divisão por zero.
4. **NumberFormatException**: Essa exceção ocorre quando você tenta converter uma string em um tipo numérico, mas a string não está no formato correto para a conversão. Por exemplo, tentar converter a string "abc" para um número.
5. **FileNotFoundException**: Essa exceção é lançada quando você tenta acessar um arquivo que não existe no sistema de arquivos.
6. **IOException**: Essa é a exceção base para problemas de entrada e saída. Ela cobre uma ampla variedade de exceções que podem ocorrer durante operações de entrada/saída, como leitura ou gravação de arquivos.
7. **ClassNotFoundException**: Essa exceção ocorre quando você tenta carregar uma classe usando Class.forName(), mas a classe especificada não pode ser encontrada no classpath.
8. **IllegalArgumentException**: Essa exceção é lançada quando um argumento passado para um método não é válido. Isso pode ocorrer quando os valores dos argumentos estão fora do intervalo esperado ou quando não atendem a certas condições.
9. **InterruptedException**: Essa exceção é lançada quando um thread é interrompido por outro thread enquanto está esperando ou em execução. Geralmente está relacionada a operações de multithreading.
10. **RuntimeException** (e subclasses): Essa é uma classe base para exceções que ocorrem durante a execução de um programa, mas não são verificadas em tempo de compilação. Isso inclui exceções como NullPointerException, ArrayIndexOutOfBoundsException e outras mencionadas acima.
11. Lidar com exceções é uma parte importante da programação Java para garantir que seu código seja robusto e possa lidar com situações inesperadas de maneira controlada. Isso é geralmente feito usando blocos try-catch para capturar e tratar as exceções, ou declarando que seu método pode lançar determinadas exceções usando a palavra-chave throws.